Zend_Dojo - Extended Dijit Support - Matthew Weier O'Phinney

<ac:macro ac:name="unmigrated-inline-wiki-markup"><ac:plain-text-body><![CDATA[

<ac:macro ac:name="unmigrated-inline-wiki-markup"><ac:plain-text-body><![CDATA[

Zend Framework: Zend_Dojo – Extended Dijit Support Component Proposal

Proposed Component Name	Zend_Dojo – Extended Dijit Support
Developer Notes	http://framework.zend.com/wiki/display/ZFDEV/Zend_Dojo – Extended Dijit Support
Proposers	Matthew Weier O'Phinney
Zend Liaison	Ralph Schindler
Revision	1.0 - 14 January 2009: Initial Draft. (wiki revision: 7)

Table of Contents

1. Overview
2. References
3. Component Requirements, Constraints, and Acceptance Criteria
4. Dependencies on Other Framework Components
5. Theory of Operation
6. Milestones / Tasks
7. Class Index
8. Use Cases
Creating menus using accessors
Creating a dialog
Creating a tooltip
Creating a tooltip dialog
Rendering a tree
Creating a TitlePane
Creating progress bars
Color palettes
9. Class Skeletons

1. Overview

Zend_Dojo was introduced in ZF 1.6.0, and offers support for most form dijits and all layout dijits. A number of dijits are still missing, and the goal of this proposal is to add support for the majority of them.

2. References

- Dijit API docs
- DojoCampus explorer
- Zend_Dojo documentation

3. Component Requirements, Constraints, and Acceptance Criteria

- This proposal **will** add support for the following top-level dijits:
  - dijit.ColorPalette
  - dijit.Dialog
  - dijit.Menu (and implicitly MenuItem and MenuSeparator)
  - dijit.ProgressBar
  - dijit.TitlePane
  - dijit.Tooltip
  - dijit.TooltipDialog
  - dijit.Tree
- This proposal **will** add support for the following form dijits:
  - dijit.form.MultiSelect
- This proposal **will NOT** add support for:
  - dijit.form.ComboButton
  - dijit.form.DropDownButton
  - dijit.form.MappedTextBox
  - dijit.form.RangeBoundTextBox

4. Dependencies on Other Framework Components

- Zend_Dojo

5. Theory of Operation

In all cases, the various dijits will follow the conventions and API of existing dijits supported by Zend_Dojo, including view helpers and form elements and decorators (where applicable).

In the case of dijit.Menu, the view helper will return an instance of itself, and allow the developer to programmatically add menu items and separators.

We are omitting support for several dijits that we will re-evaulate in the future. Amongst these are dijit.Toolbar, dijit.form.ComboButton, and dijit.form.DropDownButton. In these particular cases, functionality developed for this proposal will act as a pre-requisite for supporting these items in the future(in particular, dijit.Menu support and dijit.TooltipDialog).

6. Milestones / Tasks

- Milestone 1: [DONE] Identify dijits for which to add support
- Milestone 2: Create proposal
- Milestone 3: Create master issue and sub issues to track development of each dijit
- Milestone 4: Dijit development (including unit tests and documentation)

7. Class Index

- Zend_Dojo_View_Helper_:
  - MultiSelect
  - ColorPalette
  - Dialog
  - Menu

        - ProgressBar
        - TitlePane
        - Tooltip
        - TooltipDialog
        - Tree
- Zend_Dojo_Form_:
    - Element_:
        - MultiSelect
    - Decorator_:
        - TitlePane
        - Dialog
        - Tooltip
        - TooltipDialog

8. Use Cases

NOTE: In all cases, "$this" is a Zend_View instance within a view script.

UC-01

Creating menus using accessors

```
$menu = $this->menu();
$menu->addItem('Cut', array(
         'iconClass' => array('dijitEditorIcon', 'dijitEditorIconCut')
         'onClick'   => 'foo.cut();',
     ))
     ->addItem('Copy', array(
         'iconClass' => array('dijitEditorIcon', 'dijitEditorIconCopy')
         'onClick'   => 'foo.copy();',
     ))
     ->addSeparator()
     ->addItem('Paste', array(
         'disabled'  => true,
         'iconClass' => array('dijitEditorIcon', 'dijitEditorIconCopy')
         'onClick'   => 'foo.paste();',
     ));
echo $menu;
```

UC-02

Creating a dialog

```
echo $this->dialog('myDialog', $contentForDialog, array(
    'title' => 'My Dialog',
));
```

UC-03

Creating a tooltip

dijit.Tooltip expects a "label" attribute which contains the text of the tooltip. For consistency in the Zend_Dojo API, we pass this as the "content" argument (the second argument).

```
<?php echo $this->tooltip('tooltip', $tooltipContent, array(
    'connectId' => 'foo',
)); ?>
<div id="foo">This will have a tooltip</div>
```

UC-04

Creating a tooltip dialog

```
echo $this->tooltipDialog('tooltipDialog', $contentForDialog, array(
    'title'     => 'My Tooltip Dialog',
    'showDelay' => 100,
));
```

UC-05

Rendering a tree

dijit.Tree is a dojo.data-powered widget, and as such has relatively little setup other than indicating the dojo.data store object that it will populate itself from.

We *may* add support for creating dojo.data stores and models; please comment if you would like to see this support.

```
<div dojoType="dojo.data.ItemFileReadStore" jsId="treeStore"
  url="/api/tree/countries.json"></div>
<div dojoType="dijit.tree.ForestStoreModel" jsId="treeModel"
  store="treeStore" query="{type:'leaf'}"
  rootId="treeRoot" rootLabel="Leaves" childrenAttrs="children"></div>
<?php echo $this->tree('myTree', '', array(
    'model'       => 'treeModel',
    'openOnClick' => true,
)); ?>
```

UC-06

Creating a TitlePane

dijit.TitlePane is simply a content container with a title. You can click on the title bar to open or close the container.

```
// from a view script:
echo $this->titlePane('foo', $containerContent, array(
    'title' => 'I can collapse and expand',
));

// As a form decorator:
$subForm->setDecorators(array(
    'FormElements',
    array('HtmlTag', array('tag' => 'dl')),
    array('TitlePane', array('title' => 'More options')),
));
```

UC-07

Creating progress bars

While the proposed view helper can create the ProgressBar dijit, it will still be up to the developer to programatically interact with it.

By default, the id (first argument) will also be used as the jsId of the dijit.

```
echo $this->progressBar('downloadProgress', '', array('style' => 'width: 300px'));
```

UC-08

Color palettes

One unanswered question is if, by default, this should update a hidden form element; please leave a comment if you have an opinion.

```
echo $this->colorPalette('colors', '', array(
    'palette'  => '7x10', // rows and columns
    'onChange' => 'function(val) \{ alert(val); \}',
));
```

9. Class Skeletons

```
TBD; will mostly follow conventions already established in Zend_Dojo.
```

]]></ac:plain-text-body></ac:macro>
]]></ac:plain-text-body></ac:macro>